OPEN ACCESS

University of BRISTOL

Li, L., Beach, M., Nejabati, R., & Simeonidou, D. (2019). *Building SDN Agent for Wireless Local Area Networks*. Paper presented at 2019 IEEE Wireless Communications and Networking Conference (WCNC), Marrakech, Morocco.

Peer reviewed version

Link to publication record in Explore Bristol Research
PDF-document

University of Bristol - Explore Bristol Research
General rights

This document is made available in accordance with publisher policies. Please cite only the published version using the reference above. Full terms of use are available:
http://www.bristol.ac.uk/red/research-policy/pure/user-guides/ebr-terms/

Building SDN Agent for Wireless Local Area Networks

Li Li, Mark Beach
Communications Systems and Networks Group
University of Bristol
Bristol, United Kingdom
{l.li, m.a.beach}@bristol.ac.uk

Reza Nejabati, Dimitra Simeonidou
High Performance Networks Group
University of Bristol
Bristol, United Kingdom
{reza.nejabati, dimitra.simeonidou}@bristol.ac.uk

***Abstract*—Recently, a lot of research is focused on applying Software-Defined Networking (SDN) concepts on wireless networks. However, the current wireless systems are lack of SDN support and remain closed and (mostly) proprietary, which makes it difficult to integrate with SDN systems. The SDN agent is introduced to solve the above issues. An SDN agent is a software element bridging the SDN controller and any legacy wireless network elements (NEs) by providing the abstraction of these elements. The main advantage of this approach is that there is no modification to either the existing SDN elements or the 802.11 protocols, while the SDN controller is enabled to control and manage legacy wireless NEs. In this paper, we present an SDN agent framework for WLANs and describe an implementation of the proposed agent on the Wireless Open-Access Research Platform (WARP), an FPGA based open source Software-Defined Radio (SDR) platform. With the support from such an agent, innovative applications and functionalities can be developed for the WLAN access points (APs), such as dynamically slicing AP bandwidth among users and adapting the transmit power in flexible granularity, i.e. at per frame/per flow level.**

Keywords— Software-Defined Networking, IEEE 802.11, SDN agent, slicing, transmit power control, Software-Defined Radio

I. INTRODUCTION

Wireless local area networks (WLANs) have become more and more popular and ubiquitous due to its ability to provide unrestricted connectivity and mobile access. The family of IEEE 802.11 standards, commonly known as Wi-Fi, has been widely used in residential, commercial and industrial environments as well as on various devices, including access points (APs), mobile phones, laptops and TVs, etc. There is an ever increasing demand for higher capacity, higher data rate and higher throughput, and this is accelerated by recently emerging bandwidth hungry applications such as online gaming, high-definition multimedia streaming, Internet of Things (IoT) [1] and mobile data offloading [2].

At the same time, emerging dense WLAN scenarios, such as stadiums and public transportations, require the deployment of many APs in close range of each other in a bid to offer sufficient services to many stations (STA). However, if APs are not carefully managed, the throughput performance of the network can be significantly degraded due to the link suppression and deadlock effects [3].

Therefore, for the next generation of WLANs, it is a fundamental challenge to manage, optimize, and coordinate all the devices in dense environments, which demands a new control mechanism with more flexibility, scalability and effectiveness.

In Software-Defined Networking (SDN) paradigm [4], by separating the control plane from the data plane, the controller makes the network forwarding decisions and directly controls the network to achieve desired operating behaviour, hence enhancing the network control programmability. SDN provides a simplified and improved way for telecom software developers to design and develop mechanisms for controlling network resource. Recently, SDN principles have been used in wireless networks including WLANs and cellular networks, not in wired networks domain only.

However, the current wireless systems are lack of SDN support and remain closed and (mostly) proprietary. Most commercial WLAN controllers only work with compatible APs, mostly from the same manufacturer. Besides, WLAN device manufacturers are reluctant to provide APIs for third-party application developers. Therefore, it is almost impossible for the SDN controller to control and manage the underlying wireless Network elements (NEs).

In this paper, a framework based on an SDN agent is introduced to solve the above issues. An SDN agent is a software element bridging the SDN controller and any legacy wireless network elements by providing the abstraction of these elements. To demonstrate the capability of such an agent, the SDN Wi-Fi agent has been implemented on Wireless Open-Access Research Platform (WARP) [5]. This SDR platform not only enables flexible implementation of PHY and MAC protocols via software, but also provides the information of PHY and MAC parameters and states. It is worth noting that the SDR platform is not a necessity to our SDN agent approach, but rather a simple and practical way for prototyping the concept. The agent uses the same concept of middleware and hides the details of underlying NEs, while NEs decide what to disclose and how to.

The SDN agent facilitates the WLAN APs to integrate with SDN domain by providing the abstraction of WLAN resources. As in the wired networks, innovative applications and new functionalities can be developed for the SDN agent-based APs, leading to a better WLAN management and operation. Our approach does not require changes to SDN and WLAN protocols, which can serve as a good starting point for the wireless network virtualization and the convergence of all wireless networks.

In summary, our main contributions in this work are:

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

- Design and development of an SDN agent framework, a novel approach to bring legacy WLANs into SDN domain
- Design and implementation of the proposed framework on WARP 802.11 platform, a highly configurable SDR system
- Provision of proof-of-concept experiments to showcase the capability of the SDN agent approach, including dynamic bandwidth slicing in a Wi-Fi AP and transmit power control at per packet/per flow level.

The rest of the paper is organized as follows. Section II provides the brief background information on the SDN agent and the WARP platform. Section III describes the design of the SDN WLAN agent and the implementation on the WARP platform. In Section IV, experiments are presented to show the capability of the SDN agent approach, followed by Section V with conclusion and future work.

II. Background Information

In this section, we overview the basic concepts of SDN and describe briefly WARP SDR platform used in our implementation.

A. Basic Principles of Software-Defined Networking

According to the Open Networking Foundation (ONF) white paper [4], the SDN architecture is comprised of application, control and data plane. Shown in Fig.1, the data plane consists of network elements with the Control-Data-Plane Interface (CDPI) agent to expose the capabilities of SDN datapaths to the controller. In order to do this, the CDPI agent is responsible for carrying out the commands of the controller and notifying the controller of events that are specified by the controller. It is also noted that in the control plane there are agents called Northbound Interfaces (NBI) agents, which are interfaces between SDN controllers and SDN applications. In this paper, we focus on the agents in the data plane, while the NBI agents are out of scope of this paper.

The data plane contains one or more than one network elements, each of which has a set of forwarding or traffic processing resources. While the data plane agents support the concept of sharing or virtualizing the underlying physical resources of NE, resources are always abstractions of underlying physical capabilities or entities.

Also shown in Fig.1, the main components of NE resources block [6] are:

- Data pipeline, including data sources and data sinks, where the traffics entering and leaving the NE respectively.
- Forwarding and/or traffic processing engines, such as QoS, filtering, monitoring or tapping. Those function blocks decide how the traffic flows are to be handled.
- A virtualizer whose function is to expose the resources abstraction to the SDN controller and enforce policies.

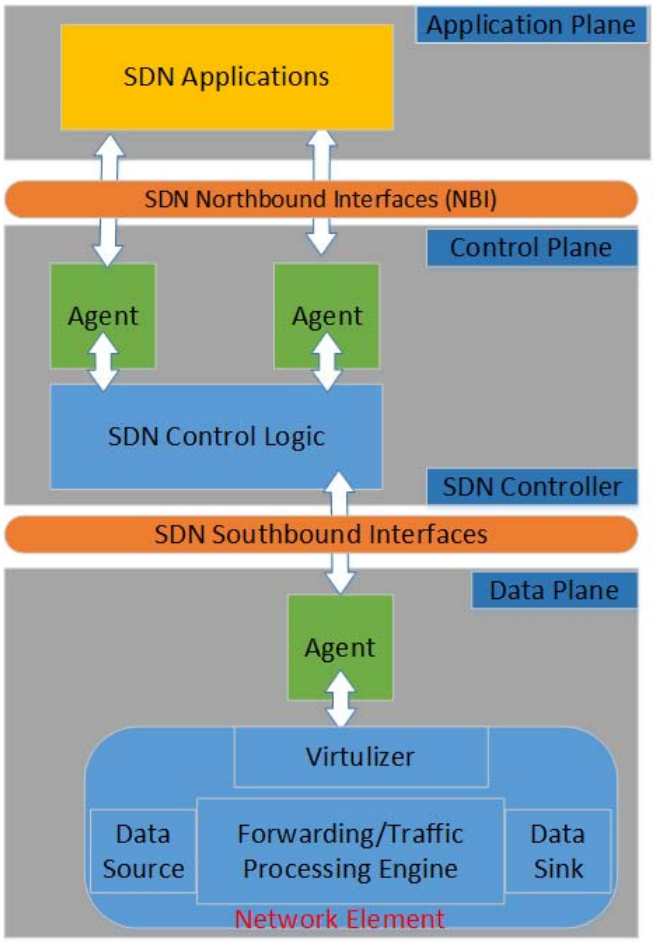

Fig. 1. Overview of SDN architecture [6]

B. Wireless Open-Access Research Platform

Wireless open-Access Research Platform (WARP) [5] is a highly programmable, scalable and extensible wireless platform. The platform is comprised of both custom hardware and field-programmable gate array (FPGA) implementations of key communication blocks. The hardware consists of FPGA-based (Xilinx Virtex-6) processing boards, wideband radios and other I/O interfaces. The Mango Communications 802.11 Reference Design [7] can interoperate with commercial 802.11 devices, operating as an Access Point (AP), client (STA) or ad-hoc node (IBSS). In this paper, we focus on an SDN agent designed and developed for 802.11 APs based on the WARP reference design.

The WARP 802.11 Reference Design is a complete FPGA implementation of IEEE 802.11 MAC and PHY. The primary MAC functionalities are realized in software, running in two MicroBlaze CPUs [8] with an FPGA core to guarantee accurate inter-packet timing. Multiple custom FPGA cores are used to implement 802.11 PHY.

Both the hardware specifications and algorithm implementations are freely available via the online open-access repository. A complete FPGA architecture of WARP 802.11 reference design can be found in [9].

C. Related Work

Odin [10] is an SDN Wi-Fi network architecture based on the concept of light virtual AP (LVAP), which separates states of each station association from physical APs. LVAPs are located on Odin agents that operate on APs and are implemented in the Click Modular Router [11]. In [12], an SDN framework is presented to extend basic concept of SDN to WLANs. AEtherFlow is implemented in CPqD SoftSwitch [13], which is installed on OpenWRT [14]. Both Odin and AEtherFlow are implemented on commodity AP hardware, which results in the limited exposure of the underlying network resources. This, in turn, restricts the support of programmability in these models. Therefore, their applications are mostly focused on client-AP authentication and association related scenarios, while our approach allows the control of layer-2 and layer-1 WLAN parameters.

The authors of [12] also present a cross-layer architecture using the principles of SDR and basic building blocks of SDN. In its implementation, GUN radio [16] is used on a Universal Software Radio Peripheral (USRP) platform. CrossFlow [15] is a general SDN framework and does not specify how it can be integrated into 802.11 stack. All aforementioned approaches more or less rely on OpenFlow switches in their systems, i.e. some of SDN functionalities and applications are only supported by OpenFlow switches. Our SDN agent approach does not require the assist from SDN switches mandatorily.

III. Implementation of SDN Agent on WARP

The principles of abstracting network resources and states to SDN applications are the foundations for the programmability of the network [4]. Based on the information from the NE, SDN applications are enabled to specify requirements and request changes to their network services via the SDN controller, and to programmatically react to network states. The above principles require that the SDN controller has a certain degree of control on the SDN datapaths, and the data plane agent is the exact entity that can fulfil such requirements by executing the SDN controller's instructions in the data plane.

There are two main functionalities in the SDN 802.11 agent:

- Firstly, providing a two-way communication with the SDN controller, including receiving and processing messages from the SDN controller, constructing and sending response to the SDN controller, allowing a controller to query the related statistics of the underlying wireless networks.
- Secondly, controlling the 802.11 MAC and PHY behaviours according to the instructions from the SDN controller in real-time. This covers not only higher MAC functionalities, such as authentication, association, channel assignment, SSID configuration, etc., but also the lower MAC tasks, such as the frame transmissions and receptions

To achieve the above goals, the SDN agent must have the ability to control the MAC and PHY functionalities of 802.11 datapath. In other words, the SDN agent must have the capability to affect the forwarding and traffic processing behaviour of 802.11 data plane. Therefore, the SDN agent enables the 802.11 datapath programmability, which in turn allows the SDN applications to set specific transmission settings at per flow or per frame level on WARP AP.

A. Communication Mechanism

As defined in [6], each interface in the SDN architecture is implemented in a driver-agent pair. On the SDN controller side, a driver for the SDN WLAN agent is developed to command construction and response processing. Also on the SDN agent side, the command parsing and the response generation functions are implemented accordingly.

For practical reasons, we build our southbound protocol by extending message system of the WARP experiments framework [5]. A message is basically constructed with a message ID and a list of arguments. For each message, a message handler is implemented in the SDN agent to execute the instructions issued by the SDN controller. Normally, there are two types of messages. One is the action command which intends to alter the behaviours of the datapath in NE, while the other is the information retrieve command which requests states of NE and will have no impact on the 802.11 MAC and PHY behaviours.

The instructions are supplied within a list of command arguments constructed as: [ARG0, ARG1, … , ARGn] and the arguments would contain the following fields:

- Matching fields: layer2-layer4 fields defined in OpenFlow standard
- Action types: write, update, or read
- Action parameter pairs: <key, value>

B. SDN WLAN Agent Model

Messages with instructions are sent from the SDN controller to the agent in the data plane, where the messages are parsed. Once the SDN agent interprets the command, it will create a filter in high-level MAC, right at the start of the transmission (Tx) datapath. The filter itself contains the match condition and actions stated in the command message.

When a new data packet enters the Wi-Fi NE, mostly from an Ethernet interface, the WARP framework will create a metadata attached to this new packet. The metadata contains the description of how this packet should be dealt with when it is going through each stage of the 802.11 datapath. Most of the metadata are parameters associated with the transmission, i.e. how this packet is going to be transmitted by the Wi-Fi interface.

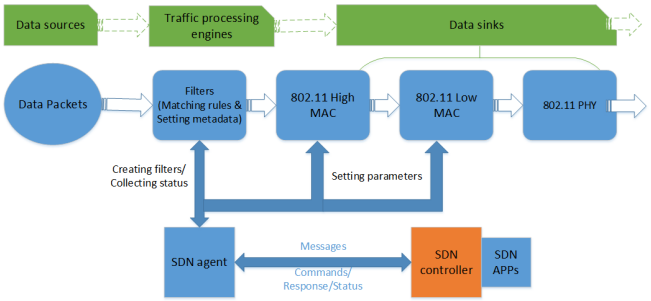

Fig. 2. Interactions among the SDN agent, 802.11 datapath and SDN controller

Fig.2 illustrates how the proposed SDN agent provides the WLAN transmission control for the data entering the

WLAN NE. Every new data packet will enter a chain of filters created by the SDN agent. When the packet matches a specified filter, its metadata will be modified accordingly, indicating how this packet will be processed through the 802.11 datapath. Fig.2 also depicts the resource blocks in a generic NE and their corresponding functionalities in the WARP 802.11 Tx datapath.

As a proof-of-concept, we implement a very simple packet-processing module with basic matching and action functionalities in the SDN agent at this stage of our work. The implementation of more complex and more powerful data model will be discussed in Section V.

C. *Implementation*

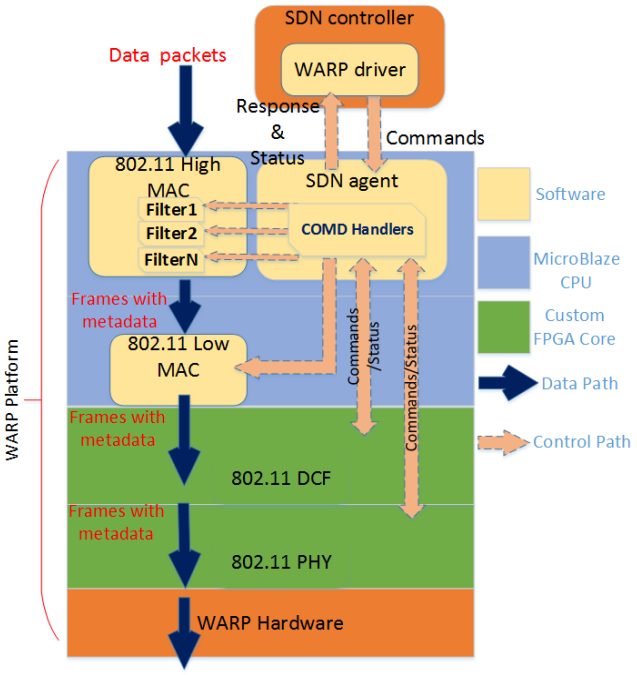

Fig. 3. SDN WLAN agent on WARP

The SDN WLAN agent is developed in C on MicroBlaze CPU of WARP and sits right on top of 802.11 AP high MAC, which means that the added agent functions will not affect the performance of lower-layer Tx/Rx tasks. The WLAN driver on the SDN controller side is implemented in Python. Fig.3 depicts the architecture view of the SDN agent implementation. The SDN agent implements a set of rules/polices in form of filters at the start of Tx datapath, acting like traffic processing engines. When a new packet enters the Tx datapath, the traffic processing engines try to check if the packet matches a specified rule set by the SDN agent. If there is a match, the pre-defined action will be taken, such as setting/modifying a parameter in the metadata of this packet. After leaving the processing engines, the packet with its metadata will then pass through 802.11 high MAC, low MAC and PHY and finally will be transmitted to the medium. At each of these stages, the parameter(s) in the metadata will be checked and applied when necessary.

In addition to modifying metadata of the frame, the SDN agent can directly access parameters of MAC and PHY modules, hence allowing the SDN controller to control these modules as well, shown in Fig.3.

IV. EXPERIMENTS

A. *Experiments Setup*

Our test-bed uses a simple network topology, as shown in Fig.4. It is comprised of a PC hosting an SDN controller and a traffic generator, a WARP board acting as AP, and two off-the-shelf 802.11 clients (STA). As described in Section III, an SDN WLAN agent is implemented on the WARP board and communicates with Ryu [17] SDN controller. In this test-bed, two separate Ethernet cables are used for the communications between the PC and WARP AP, one for the control path between the SDN controller and the SDN agent, and the other for the data path between the traffic generator and AP.

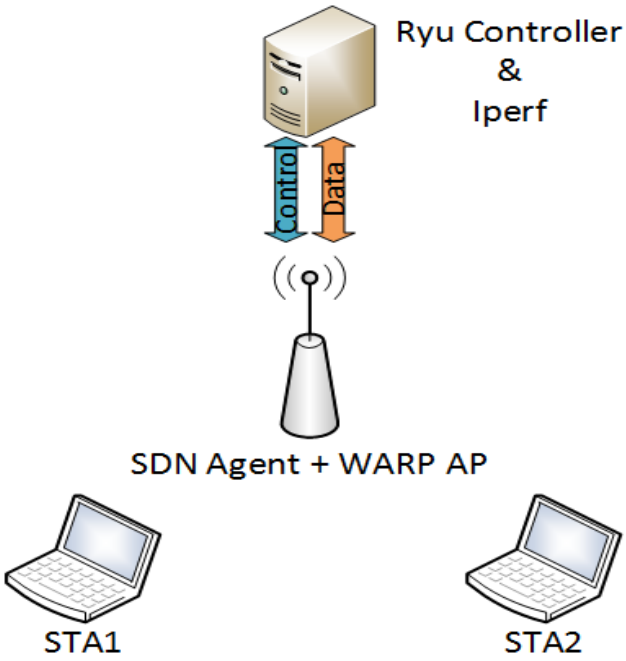

Fig. 4. Experiment topology

B. *WLAN Access Point Bandwidth Slicing*

The WARP 802.11 reference design has a Tx queue for each STA associated with the AP. When the low-level MAC is ready for a new packet, the high-level MAC removes the head packet from the next available queue and passes it to the low-level MAC for transmission. By default, the next queue is selected via round robin (RR), therefore, the traffic for each station has the same chance to be delivered, hence the bandwidth of AP being shared equally among STAs. Please note that only the downlink traffic is considered in this scenario.

A weighted round robin (WRR) queue management is implemented to replace the default RR. Therefore, each STA can have different shares of the AP bandwidth according to the weight associated with its queue. With the assist from the SDN agent, the weight of each queue can also be dynamically set or changed by the SDN controller on the fly. The AP will now serve the data traffic of an STA based on the weight value assigned to this client.

Fig.5 illustrates the work flow of the SDN WLAN agent in this experiment. The commands issued by the SDN controller contains the station information (STA_Info) which

helps the SDN agent to locate the corresponding queue of a client. In this case, the SDN agent sets the weight directly to Tx queue of each STA, instead of creating any filters.

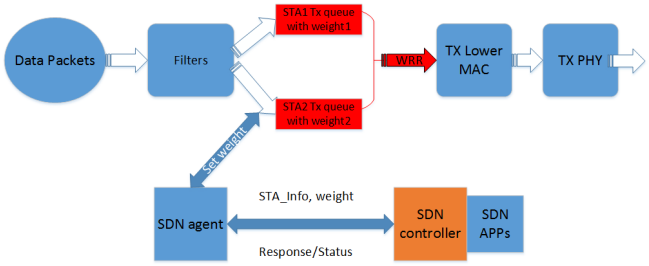

Fig. 5. Slicing AP bandwidth with SDN agent

During the experiment, two identical UDP traffics are generated for STA1 and STA2 at the same time. At the beginning of the experiment, because the queue for each STA has the same weight on AP, each STA has half of the bandwidth of AP, around 7.8 Mbps, as shown in Fig.6. At about 55 seconds, the SDN controller sets the weights of the two STAs as 2 and 1 respectively, which indicates that the network bandwidth of AP should be allocated in 2:1 ratio between the two clients. It can be observed that STA2 has about 10.8 Mbps throughput, about 66% of the available bandwidth, while STA1 obtains a throughput of 5 Mbps, i.e. 33% of the AP bandwidth. At about 125 seconds, the SDN controller swaps the weights of the two STAs. We can see that STA1 has about 10.8 Mbps bandwidth while STA2 has about 5.0 Mbps. In this way, the network bandwidth of AP is dynamically sliced.

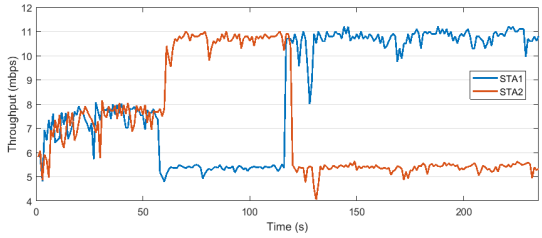

Fig. 6. STA throughput vs. time

The main purpose of this experiment is to show that new functionalities can be simply added to the SDN-enabled AP. More sophisticated queue management schemes can be implemented based on the above approach. It also demonstrates that the SDN agent can provide enough visibility of network resources without exposing excessive details. A bandwidth allocation application only needs to define how much bandwidth for a STA without knowing the details of how WLAN AP is going to achieve this.

C. *Fine-grained Transmit Power Control*

In dense WLAN scenarios, in order to increase spatial reuse as well as reduce interference, transmit power control (TPC) mechanisms are proposed in IEEE 802.11ax [18]. So far, the dynamic adaptation of transmit power is standardized at per link/per user level. In this experiment, we demonstrate that our approach can achieve dynamic transmit power control at per packet, per flow in addition to per link level.

Three UDP traffics are generated to STA1. The SDN controller decides to use different transmit power for individual traffic, in this case, 2 dBm for flow1, 4 dBm for flow2 and 8 dBm for flow3. The flows can be differentiated by their destination IP addresses and UDP port numbers. Filters (as listed below) are created in WARP AP by the SDN agent.

TABLE I. FILTERS CREATED BY THE SDN AGENT

Match Fields		Actions
IP Dst Address	***UDP Port***	
192.168.1.10	50001	Setting tx_power to 1
192.168.1.10	50002	Setting tx_power to 2
192.168.1.10	50003	Setting tx_power to 3

Once the packet of a flow is matched in the filters, the defined Tx power will be assigned to its metadata. When this frame reaches WLAN interface, it will be transmitted with the assigned Tx power. A spectrum analyser is used to measure the strength of the transmitted signals.

Fig.7 depicts 10 seconds of the measurement results captured by the analyser and shows that there are signals with 3 different strength levels.

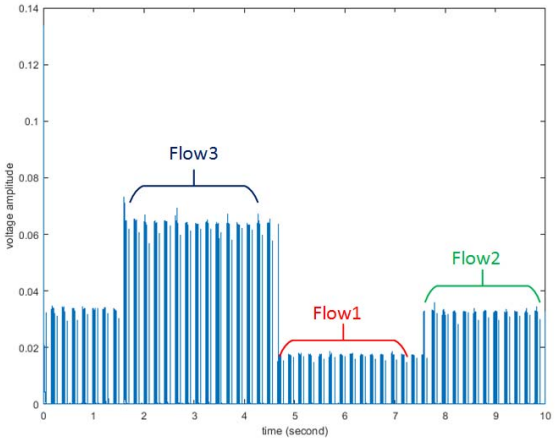

Fig. 7. A snapshot of WLAN transmissions with different Tx powers

The SDN controller can also set or change the transmit power of on-going transmissions on the fly by issuing commands to the agent, and the agent will adjust the corresponding filters accordingly. Therefore, the Tx power for each user, each flow or even each packet can be changed dynamically.

V. CONCLUSION

In this paper, we presented an SDN agent framework for the management and control of legacy WLANs. The proposed approach provides a greater degree of network programmability than the current Software-Defined WLAN (SDWLAN) solutions, which allows a simple way to develop innovative applications and implement new functionalities for agent-based APs.

The main features of our SDN WLAN agent framework are listed below:

- No modification to the SDN elements and 802.11 protocols

- Better layer-2 and layer-1 visibility of 802.11 protocol stack to SDN controller without disclosing excessive details
- Real-time control of 802.11 MAC and PHY behaviours via SDN controller without interfering the real-time operation of Wi-Fi interfaces

Since the traffic-processing module is the key element of the data plane, our future task would focus on enhancing the packet–processing functionalities in the SDN agent. Instead of using software filters, we plan to develop an enhanced packet-processing module on an FPGA core in WARP. We would also extend our framework to LTE/LTE-A networks by designing LTE agent. The results of our work will provide a promising way to the convergence of all wireless networks.

ACKNOWLEDGMENT

The authors would like to thank Dr. John Harris for the help on WARP platform. This work is supported by EPSRC grand EP/L020009/1: Towards Ultimate Convergence of All Networks (TOUCAN).

REFERENCES

[1] S. Tozlu, M. Senel, W. Mao, and A. Keshavarzian, "Wi-Fi enabled sensors for internet of things: A practical approach," IEEE Communications Magazine, vol. 50, no. 6, pp. 134–143, 2012.

[2] Y. He, M. Chen, B. Ge, and M. Guizani, "On wifi offloading inheterogeneous networks: Various incentives and trade-off strategies," IEEE Communications Surveys & Tutorials, vol. 18, no. 4, pp. 2345–2385, 2016.

[3] Z. Zhong, P. Kulkarni, F. Cao, Z. Fan, and S. Armour "Issues and challenges in dense wifi networks" Wireless Communications and Mobile Computing Conference (IWCMC) 2015 International pp. 947-951, 2015.

[4] Open Networking Foundation. "Software-Defined Networking: The New Norm for Networks," 2012. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[5] K. Amiri, Y. Sun, P. Murphy, C. Hunter, J. R. Cavallaro, and A. Sabharwal,''WARP, a unified wireless network testbed for education and research,''in Proc. IEEE Int. Conf. Microelectron. Syst. Edu., Jun. 2007,pp. 53–54.

[6] Open Networking Foundation. "SDN Architecture 1.0," 2014. [Online]. Available: https://www.opennetworking.org/wp-content/uploads/2013/02/TR_SDN_ARCH_1.0_06062014.pdf

[7] "802.11 Reference Design for WARP v3". [Online]. Available: https://warpproject.org/trac/wiki/802.11

[8] "MicroBlaze". [Online]. Available: https://www.xilinx.com/products/design-tools/microblaze.html

[9] "WARP project". [Online]. Available: https://warpproject.org

[10] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, and T. Vazao,"Towards programmable enterprise wlans with odin," in HotSDN,August 2012, pp. 115–120.

[11] Eddie Kohler , Robert Morris , Benjie Chen , John Jannotti , M. Frans Kaashoek, The click modular router, ACM Transactions on Computer Systems (TOCS), v.18 n.3, p.263-297, Aug. 2000.

[12] M. Yan, J. Casey, P. Shome, A. Sprintson, and A. Sutton, "AEtherflow:Principled wireless support in sdn," in IEEE 23rd International Conference on Network Protocols (ICNP), 2015, pp. 432–437.

[13] "CPqD SoftSwitch." [Online]. Available: https://github.com/CPqD/ofsoftswitch13

[14] "OpenWRT." [Online]. Available: https://openwrt.org

[15] P. Shome, M. Yan, S. M. Najafabad, N. Mastronarde, A. Sprintson, "CrossFlow: A cross-layer architecture for SDR using SDN principles", Proceedings of the IEEE NFV-SDN Conference, pp. 37-39, November 2015.

[16] "GNU Radio". [Online]. Available: http://enuradio.org/redmine/projects/anuradio/wiki

[17] "Ryu SDN Framework." [Online]. Available: http://osrg.github.io/ryu

[18] B. Bellalta, "IEEE 802.11ax: High-efficiency WLANs" IEEE Wireless Commun. Mag. vol. 23 no. 1 pp. 38-46 Feb. 2016.